Position-Based Permutations

Ross Ure Anderson *

27th January, 2026

Abstract. This article gives a development from first principles of some core properties of position-based permutations including permutation notations, group structure, relationship to symmetric group of degree n, inverse permutations, cyclic permutations, transpositions, parity and signature of permutations, and permutation extensions. It provides a sufficient basis for development of $n \times n$ matrix inverses and determinants [15]. It contrasts position-based permutations with value-based permutations and gives a critique of value-based permutations, including a discussion of anti-groups and anti-isomorphisms. A theorem is provided on permutation extension (§12) that has application in proving the equivalence of the Leibniz and Laplace forms of the determinant of an $n \times n$ matrix over a general field [15, Theorem 32, §12.4].

*E-mail: ruanderson100@yahoo.com, Web: https://archive.org/details/@ross_ure_anderson

Contents

1 Introduction **3**

2 Cauchy Two-Line Notation For Permutations **5**
2.1 Cauchy One-Line Notation 7

3 Value-Based Permutations **8**
3.1 Translating Between VBP's And PBP's 9

4 Compositions Of Permutations **10**

5 Permutations As A Group **10**
5.1 Group Structure For VBP's 11

6 Inverse Permutations **11**

7 Cyclic Permutations **12**

8 Transpositions **13**

9 Parity Of A Permutation **14**

10 Signature Of A Permutation **17**

11 Calculating Parity And Signature Of A Function in S_n **17**

12 Extending A Permutation **18**

13 PERM(n) Versus S_n **19**

Appendices **20**

Appendix A Opposite Groups And Anti-Isomorphisms **20**

References **22**

1 Introduction

The following notations will be used :

- $\mathbb{N}$ = the set of natural numbers $\{1, 2, 3, \ldots\}$
- $\mathbb{W}$ = the set of whole numbers (ie. non-negative integers), $\{0, 1, 2, 3, \ldots\}$
- $[n]$ = the set $\{1, 2, \ldots, n\}$
- PBP = position-based permutation of a list of n objects
- VBP = value-based permutation of a list of n objects
- PERM(n) = the set of all permutations of a list of n objects
- S_n = the symmetric group of degree n consisting of all bijections from $[n]$ to $[n]$
- $\begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$ = Cauchy two-line notation for permutations
- $\begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$ = Cauchy one-line notation for permutations
- $(r_1 r_2 \ldots r_k)$ = k-cycle permutation
- AT = adjacent transposition, ie. a 2-cycle, which swaps two adjacent positions
- PAR = $\{\text{even}, \text{odd}\}$ = the set of parities
- par = parity function (applies to $\mathbb{W}$, PERM(n), and S_n)
- G^{op} = opposite group (aka anti-group) of $(G, \cdot)$, with the binary operation reversed
- $G \sim H$ = groups G and H are isomorphic
- $G \overset{a}{\sim} H$ = groups G and H are anti-isomorphic
- iff = if and only if

In this article we develop some core properties of permutations sufficient for development of the theory of $n \times n$ matrix inverses and determinants [15], [16]. The type of the permutation will be a position-based permutation (PBP) as opposed to a value-based permutation (VBP). At various points (particularly §3 and §5) we will discuss the differences between the PBP and VBP and point out some disadvantages of the VBP. The word 'category' used in this document refers to the common definition of the word rather than the formal mathematical notion.

A permutation will be considered to be a reordering of a finite sequence (or list) of n objects, where n is an arbitrary positive integer. Like the idea of direction in Euclidean space [15, §2] it is a primitive notion not definable in any simpler terms — we can simply say it as 'a reordering of a list of n objects'. In particular the idea does not directly equate to the set based concept of a function, however it does relate closely to the idea of a function as we discuss below. Like other primitive concepts although we cannot further define it we can make certain assertions about it similarly to what is done with primitive notions in physics such as mass, time, force, and electric charge and so on where postulates are set down. For example we can establish that there are $n!$ permutations of n objects by a process of counting — there being n ways of selecting the first object in the reordering, $n - 1$ ways for the second, ..., 2 ways for the $(n-1)^{\text{th}}$, and 1 way for the n^{th} — thus giving $n \cdot (n-1) \cdots 2 \cdot 1 = n!$ different ways in total. The common usage of the term 'permutation' to mean a bijection on a general set X can cause some confusion, and we will not use this meaning in this document.

The objects permuted may be any kind of mathematical object, for example numbers, vectors, functions, subsets of a given set, roots of a polynomial, or rows or columns of a matrix, or indeed they may be any kind of entity at all, not just mathematical, for example they may be colours, animals, car models, or a collection of completely unalike objects. The key thing is that they are a sequence of well-defined objects that come in a specific order, so that there is a first, a second, ..., an $(n-1)^{\text{th}}$, and an n^{th}. The objects need not be unique, ie. two or more of them may have the same value. However their positions are unique, with each object having a unique position within the range $1, 2, ..., n$.

Permutations can be carried out on infinite sequences, for example in rearrangements of infinite series [3, p328-329] but we will only consider finite sequences here.

The finite sequence to which a permutation applies is similar to the formal mathematical notion of an n-tuple and we use the same notation, for example the following all denote sequences : $(1, 2, 3, 4)$, (f, g, h) (where f, g, h are functions), $([4, 7, 53, 2], [-8, 2, -1, 5], [12, -17, 25, 2])$ consisting of matrix rows, (dog, sparrow, cat, horse) consisting of animals, $(503, 43, 22, 58, 17)$, and $(3, 11, 4, 28, 3, 2, 4)$. In the last example note the values 3 and 4 are repeated.

A permutation of a sequence simply reorders its elements, the elements themselves remain the same as before, for example a permutation of $(1, 2, 3, 4)$ is $(4, 2, 3, 1)$, a permutation of (dog, sparrow, cat, horse) is (horse, cat, sparrow, dog), and a permutation of $(3, 11, 4, 28, 3, 2, 4)$ is $(11, 3, 3, 28, 4, 4, 2)$. The essence of the permutation is the manner in which the positions of the elements within the sequence are changed. The actual values of the elements do not play any part in the permutation itself. So for example given an $n \in \mathbb{N}$ $(n \geq 4)$ the following are all permutations of any sequence of n objects :

- reversal of the order of the n elements
- interchange of the order of the first two elements
- swap the first and the last element then swap the 3rd and 4th elements

The set of all permutations of a list of three objects is the following set : {identity, swap of the first two elements, swap of the last two elements, swap of the first and the third elements, rotation one place forwards, rotation one place backwards}, so that after applying these permutations to the three objects 123 we obtain the respective reorderings 123, 213, 132, 321, 312, 231.

The word 'permutation' can refer to a specific list after a permutation operation has been applied to it, as well as to a general permutation operation which can be applied to any list of the relevant length, the intended meaning will be clear from the context.

We use the notation PERM(n) to denote the set of all permutations of n objects, which is a set of size $n!$.

In many texts the word 'permutation' is used in the following quite different sense — a permutation of a set X is defined to be a bijective mapping from X to X. The set of all such bijections is then denoted S_X, which forms a group under mapping composition called the symmetric group on X [1, p71] or the full symmetric group on X [2, p8-9]. The mapping composition operation $f \circ g$ defines $(f \circ g)(x) = f(g(x))$ and $f \circ g$ is also commonly written in the product notation fg, [8]. S_X is always non-abelian for $n > 2$, [2, p8]. When $X = \{1, 2, \ldots, n\}$ the group is denoted S_n and is called the symmetric group of degree n, [1, p71], [2, p9], [7]. The key difference in this definition of permutation is that it is an operation defined on a set, whereas in our case it is an operation defined on a sequence. We will not use the set-based definition of permutation in this document but we discuss the related topic of value-based permutation (VBP) in §3 and at various other points. Otherwise the results and concepts in this document will all apply to PBP type permutations.

A PBP permutation may be represented mathematically by a function $\sigma : [n] \to [n]$, where $[n]$ denotes the set $\{1, 2, \ldots, n\}$, which specifies the position $\sigma(r)$ which the element originally at position r moves to. Due to the 'physical' meaning we are attaching to the word 'permutation' it is clear that any such function σ must be bijective (ie. one-to-one), and indeed given any arbitrary bijection $f : [n] \to [n]$ there is a unique permutation of n objects which has its associated σ equal to f. So permutations are in a one-to-one correspondence with the set of all bijections on the set $[n]$, ie. with the set S_n, so that there is a bijective mapping $\theta : \text{PERM}(n) \mapsto S_n$. We find in §5 below that θ is a group isomorphism between PERM(n) and S_n. This justifies thinking of permutations as *being* the elements of S_n, or more generally as the elements of S_X for a general set X as discussed above. In this document we make use of the one-to-one correspondence with S_n but still regard a permutation itself as being a physical process of some kind or a thought process (ie. a certain 'reordering' of a sequence of n objects), for which there is a formal mathematical representation which is the permutation function σ as described above. A VBP permutation is a member of PERM(n) just like a PBP permutation but the way its associated permutation function σ is defined is different from the PBP (see §3).

Because of the above one-to-one correspondence θ we sometimes use the notions of 'permutation' and 'permutation function' interchangeably, provided the context makes it clear which is meant. For example we might refer to a permutation in PERM(n) by its associated permutation function σ and vice-versa. It is important though to understand the distinction between the concepts of 'permutation' and 'permutation function'. 'Permutation' is the native idea, whereas 'permutation function' is the formal mathematical expression of that idea — these belong to two different categories and to conflate them as being the same is a category error. The function σ is a formal mathematical object representing the permutation as opposed to a set of instructions in words describing it or a general reordering process that is specified. When we have a formal mathematical object like σ we can perform mathematical operations and arguments with it conveniently as we do in developing the theory of $n \times n$ matrix inverses (eg. [15]). If we just worked with the original notion of permutation without the convenience and brevity of the formal mathematical notation then it would be cumbersome to express these operations within arguments. However it is important to keep in mind the native permutation ideas that underlie the formal notation.

2 Cauchy Two-Line Notation For Permutations

A permutation $P \in \text{PERM}(n)$ of n objects can be specified by the following notation, known as Cauchy's two-line notation:

$$P = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

where $a_r \in [n]\ \forall\ r \in [n]$ and the a_r's are all distinct.

The second line of the notation shows the positions of the objects in the first line after the permutation P has been applied. P is then the permutation which for each $r \in [n]$ moves the element at position a_r to position r (note a_r is in position r in the lower line and it must have appeared at position a_r in the upper line). It is necessary that all the a_r are distinct — otherwise the notation does not correctly specify a permutation. Note although we use commas in the notation for a sequence/list/n-tuple we don't use commas in the Cauchy two-line notation itself, as in matrix notation.

Because of the convention of interchanging the terms 'permutation' and 'permutation function' we can also write[1] :

$$\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

where σ is the unique permutation function associated with P. This uniquely specifies the bijective function $\sigma \in S_n$. By the definition of the PBP permutation function σ as mapping old position to new position (§1), we then have :

$$\sigma(a_r) = r,\ \forall\ r \in [n] \tag{1}$$

It follows from equation (1) that if σ is the permutation function of PBP permutation $P \in \text{PERM}(n)$ then :

$$P \quad = \quad \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma^{-1}(1) & \sigma^{-1}(2) & \cdots & \sigma^{-1}(n) \end{pmatrix}, \tag{2}$$

and we can also write :

$$\sigma \quad = \quad \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma^{-1}(1) & \sigma^{-1}(2) & \cdots & \sigma^{-1}(n) \end{pmatrix}.$$

Thus if we knew σ and wanted to determine P we would have to compute the inverse function of σ. And conversely if we knew P and wanted to calculate σ we would have to compute the inverse of the function on $[n]$ specified in the second line of the two-line notation in equation (2).

Example 1. Let $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 3 & 2 \end{pmatrix}$. Then by observation we see this denotes the permutation P described in words by :

- the element at position 1 is moved to position 2
- the element at position 2 is moved to position 5
- the element at position 3 is moved to position 4
- the element at position 4 is moved to position 1
- the element at position 5 is moved to position 3

This word description allows us to immediately write down σ since by definition σ is the mapping from old positions to new positions. Alternatively we could obtain σ from its inverse using equation (2) by reading along the bottom line looking for 1, 2, 3, 4, 5 in that order and taking the corresponding numbers above, which are 2, 5, 4, 1, 3, so that we reverse the function, obtaining $\sigma(1) = 2, \sigma(2) = 5, \sigma(3) = 4, \sigma(4) = 1, \sigma(5) = 3$, which corresponds with the word description. We do require to think slightly backwards but this is necessary for σ to be position-based as opposed to value-based. The advantages of the former over the latter will be seen in §3 below — notably the former produces a group isomorphism from PERM(n) to S_n whereas the latter produces a group anti-isomorphism from PERM(n) to S_n, §5.1, [9].

□

[1] In sources using VBP type permutations (eg. [12]) this same notation will specify a different function $\sigma : [n] \mapsto [n]$ than the one defined here — the function they specify will be our σ^{-1} (see §3.1 and §6).

In the above definition of the Cauchy two-line notation the sequence of objects being permuted was $(1, 2, 3, \ldots, n)$. However because this notation is simply a vehicle for indicating how the positions of the n objects are changed by the permutation, we can use other objects than $(1, 2, 3, \ldots, n)$, as illustrated in Examples 2 and 3 below.

Example 2. The permutation $\begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ can be denoted equally well with $\begin{pmatrix} \underline{\mathbf{u}} & \underline{\mathbf{v}} & \underline{\mathbf{w}} \\ \underline{\mathbf{w}} & \underline{\mathbf{u}} & \underline{\mathbf{v}} \end{pmatrix}$ using the vectors $(\underline{\mathbf{u}}, \underline{\mathbf{v}}, \underline{\mathbf{w}})$, or with $\begin{pmatrix} \text{dog} & \text{sparrow} & \text{cat} \\ \text{cat} & \text{dog} & \text{sparrow} \end{pmatrix}$ using the animals (dog, sparrow, cat) because the way that the positions of the objects are changed is the same in each case. To determine the permutation function σ we simply observe how the n objects in the top line have been repositioned in the bottom line. Thus each of the above three notations specify the same permutation function σ as given by $\sigma(1) = 2, \sigma(2) = 3, \sigma(3) = 1$.

□

In Example 2 above we can transcribe from any one of the three notations to another and still have the same permutation. In a practical situation we could choose the notation that best fits the problem being solved. Note though that relating P to σ using the equation (2) can only be done when the top line of the two-line notation is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$, since σ is a bijection on $[n]$. In the latter two notations in Example 2 it doesn't make sense to write $\sigma(a_r) = r \; \forall r$ since this requires that the a_r are positions, ie. numbers in the range $(1, 2, \ldots, n)$, whereas now, in the second line of the notation, we have a different kind of object. Writing $\sigma(a_r) = r \; \forall \; r$ only makes sense when the top line is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ since in this case the values of the objects are equal to their initial positions, so that in the relation $\sigma(a_r) = r$ the function σ is operating on an initial position a_r to produce the new position r. If we had a permutation specified using objects other than $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ then we could just transcribe it into a form using the objects $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ if needed. If we did that the relation $\sigma(a_r) = r$ would then make sense again. However it is convenient to be able to use objects other than $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ in the top line of the Cauchy two-line notation so that we are free to choose objects which match the problem at hand.

Example 3. Consider the following permutation σ :

$$\sigma = \begin{pmatrix} 2 & 35 & 7 & 11 & 19 & 22 & 44 & 8 \\ 44 & 35 & 22 & 19 & 11 & 8 & 7 & 2 \end{pmatrix}$$

which rearranges the list $(2, 35, 7, 11, 19, 22, 44, 8)$ into descending order $(44, 35, 22, 19, 11, 8, 7, 2)$. This is an example where we have a list of integers in the top line which are not a consecutive list of integers of the form $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$.

The permutation function σ is given by $\sigma(1) = 8, \sigma(2) = 2, \sigma(3) = 7, \sigma(4) = 5, \sigma(5) = 4, \sigma(6) = 3, \sigma(7) = 1, \sigma(8) = 6$. By transcribing the objects on the top line to $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ we can rewrite this as :

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 7 & 2 & 6 & 5 & 4 & 8 & 3 & 1 \end{pmatrix}$$

This is exactly the same permutation but the first notation is a more natural fit to the problem by showing the original list of numbers and their rearrangement into descending order.

□

Now observe what would happen if in the first notation above in Example 3 there was a repeated value in the top line, for example change the 11 to a 2 to get $\sigma = \begin{pmatrix} 2 & 35 & 7 & 2 & 19 & 22 & 44 & 8 \\ 44 & 35 & 22 & 19 & 8 & 7 & 2 & 2 \end{pmatrix}$, in which the second line again shows the correct reordering into descending order. This notation then does not specify a unique permutation σ but specifies two possible permutations, namely σ_1 given by $\sigma_1(1) = 7, \sigma_1(2) = 2, \sigma_1(3) = 6, \sigma_1(4) = 8, \sigma_1(5) = 4, \sigma_1(6) = 3, \sigma_1(7) = 1, \sigma_1(8) = 5$ (when the first 2 is mapped to position 7), and σ_2 given by $\sigma_2(1) = 8, \sigma_2(2) = 2, \sigma_2(3) = 6, \sigma_2(4) = 7, \sigma_2(5) = 4, \sigma_2(6) = 3, \sigma_2(7) = 1, \sigma_2(8) = 5$ (when the first 2 is mapped to position 8).

Thus one caveat applying to the Cauchy two-line notation is that in the choice of objects we use on the top line each object must be distinct otherwise there would be ambiguity in the definition of the permutation, eg. in $\sigma = \begin{pmatrix} \underline{\mathbf{u}} & \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{v}} \\ \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{u}} & \underline{\mathbf{v}} \end{pmatrix}$ we know position 1 goes to position 3 and position 3 goes to position 2 but there is ambiguity in where positions 2 and 4 go to due to the repetition of the object $\underline{\mathbf{v}}$.

If we were to allow duplicate objects in the top line then the Cauchy two-line notation would specify a set of possible permutations σ. If there were m distinct object values that have each been duplicated in the top line, having $l_1, l_2, \ldots, l_m$

repetitions respectively, then the notation would specify a set of $l_1! \times l_2! \times \ldots \times l_m!$ possible permutations. Such a notation could be used to denote a permutation with the understanding that one particular permutation out of the set has been selected.

Note. The above issue with repeated values only applies to the notation for *specifying* permutations. When actually applying permutations to a list there is no problem with the list containing duplicated values, since the permutation is only concerned with the positions of the objects in the list not their values.[2]

□

2.1 Cauchy One-Line Notation

When the top line in the Cauchy two-line notation is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ we can abbreviate it to the Cauchy one-line notation :

$$\sigma = \begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

where $a_r \in [n] \; \forall \; r \in [n]$ and the a_r's are all distinct. Here it is understood that this is the second line, with the top line implicitly being $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$.

For example $\sigma = \begin{pmatrix} 3 & 1 & 2 \end{pmatrix}$ means the same as $\sigma = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ and $\sigma = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$ is the identity permutation on three objects.

For this notational convention to work the numbers $\begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$ must be a rearrangement of the numbers $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ for some n since otherwise the specification of σ is ambiguous. For example the one line notation $\begin{pmatrix} 5 & 4 & 7 \end{pmatrix}$ doesn't uniquely specify a permutation since we have no consistent reason to suppose an order for the numbers $5, 4, 7$ on the omitted top line (the fact that the objects on the top line are integers does not mean that they should come in a specific order – eg. see Example 3 above).

[2] A problem does arise with duplicate values though when using VBP's — see §3.

3 Value-Based Permutations

The type of permutation we have defined above and will use throughout this article is a position-based permutation (PBP). With a PBP permutation the permutation function σ defines how the positions of the objects in the sequence are changed by the permutation. However it is possible to define a value-based permutation (VBP) where the permutation function σ is defined in a different way : σ maps each object value in the sequence to a new object value from the same sequence in such a way that a rearrangement of the sequence is produced, ie. σ maps the object values in the sequence in a one-to-one fashion, so it is a bijection on the set V of values that are in the original sequence.

In this section we contrast the VBP with the PBP since confusion often arises between these two types of permutations. Even though ultimately they both do the same thing of reordering a sequence of objects there is a difference in how the permutation function σ is defined. VBP permutations and PBP permutations are still just elements of the same set PERM(n), so they are in the same category of 'permutation' (ie. they are reorderings of n objects), but the way their associated permutation functions σ in S_n are defined is different. A discussion of VBP versus PBP can be found in the article [4].

A VBP can be defined using the same Cauchy two-line notation that we used for PBP in §2, except that σ is interpreted differently. As before we initially write:

$$P = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

where $a_r \in [n] \ \forall \ r \in [n]$ and the a_r's are all distinct, to specify the reordering which the permutation $P \in$ PERM(n) performs on n objects.

But then we write :

$$\tau = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

with a different interpretation than in §2 — the permutation function τ, for each $r \in [n]$, changes the value of the element at position r to a_r. This uniquely specifies the function $\tau : [n] \mapsto [n]$. Again all the a_r must be distinct — otherwise the notation does not correctly specify a permutation. By this definition we then have :

$$\tau(r) = a_r \ \forall \ r \in [n] \tag{3}$$

This is to be contrasted with the equation (1) in §2 for PBP's.

It follows from equation (3) that if τ is the permutation function of VBP permutation $P \in$ PERM(n) then :

$$P \ = \ \begin{pmatrix} 1 & 2 & \cdots & n \\ \tau(1) & \tau(2) & \cdots & \tau(n) \end{pmatrix}, \tag{4}$$

and we can also write :

$$\tau \ = \ \begin{pmatrix} 1 & 2 & \cdots & n \\ \tau(1) & \tau(2) & \cdots & \tau(n) \end{pmatrix}.$$

Thus if we knew τ and wanted to determine P we would just place the values of τ in the second line of the two-line notation. And conversely if we knew P and wanted to calculate τ we would just determine the values of τ on $[n]$ from the second line of the two-line notation. This is an advantage of the VBP over the PBP, since with the latter we required to compute inverse functions in order to relate P and σ, using equation (2).

For example consider the permutation $\begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ in Example 2 in §2 above. To express this as a VBP we would require τ to be the following bijection on the set $V = \{1, 2, 3\}$: $\tau(1) = 3, \tau(2) = 1, \tau(3) = 2$. Note this differs from the PBP permutation function σ we obtained in Example 2, which was given by : $\sigma(1) = 2, \sigma(2) = 3, \sigma(3) = 1$.

As in the PBP case we could have objects in the top line other than $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$, then the a_r would be drawn from those objects too.

For example with the alternate notation $\begin{pmatrix} \underline{\mathbf{u}} & \underline{\mathbf{v}} & \underline{\mathbf{w}} \\ \underline{\mathbf{w}} & \underline{\mathbf{u}} & \underline{\mathbf{v}} \end{pmatrix}$ of Example 2, using vectors as object values, the VBP would become the bijection on the set $V = \{\underline{\mathbf{u}}, \underline{\mathbf{v}}, \underline{\mathbf{w}}\}$ given by $\tau(\underline{\mathbf{u}}) = \underline{\mathbf{w}}, \tau(\underline{\mathbf{v}}) = \underline{\mathbf{u}}, \tau(\underline{\mathbf{w}}) = \underline{\mathbf{v}}$. The set V and the function τ have changed, unlike in the case of the PBP where after the notational change σ remained the same function, with the same domain $\{1, 2, 3\}$.

The VBP pays attention to the actual values of the objects being permuted, whereas the PBP is only concerned with the positions of the objects. The VBP corresponds to the idea of a permutation of a set (as opposed to a permutation of a sequence) which is defined to be a bijection on the set, these bijections forming what is called the symmetric group on the set, denoted S_V for a set V or S_n if $V = \{1, 2, \ldots, n\}$, as described in [1, p71], [2, p9], and [7].

A problem which arises with the VBP is when we require to permute a sequence which contains some repeated values. This situation can arise with matrices when we have to permute rows or columns as it is possible for there to be duplicate rows or columns in a matrix. With a PBP this situation is quite easily handled because the PBP is only concerned with positions of objects in the sequence. But with a VBP information will be lost when we pass from a sequence to a set.

For example consider the sequence $\begin{pmatrix} \underline{\mathbf{u}} & \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{v}} \end{pmatrix}$ of vectors which we want to rearrange into $\begin{pmatrix} \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{v}} & \underline{\mathbf{u}} \end{pmatrix}$ (for example these vectors may contain the rows or columns of a matrix). With a VBP we would define $V = \{\underline{\mathbf{u}}, \underline{\mathbf{v}}, \underline{\mathbf{w}}\}$ since this is the set of values in the sequence. But we encounter problems when we try to define τ. Progressing along the sequence we set $\tau(\underline{\mathbf{u}}) = \underline{\mathbf{v}}$ and $\tau(\underline{\mathbf{v}}) = \underline{\mathbf{w}}$ but when we set $\tau(\underline{\mathbf{w}}) = \underline{\mathbf{v}}$ we violate the one-to-one property of τ. But furthermore on assigning $\tau(\underline{\mathbf{v}}) = \underline{\mathbf{u}}$ we violate the function property of τ by assigning two values to $\tau(\underline{\mathbf{v}})$. With PBP's though there are two possible permutations which will perform the required rearrangement : namely σ_1 defined by $\sigma_1(1) = 4, \sigma_1(2) = 1, \sigma_1(3) = 2, \sigma_1(4) = 3$ and σ_2 defined by $\sigma_2(1) = 4, \sigma_2(2) = 3, \sigma_2(3) = 2, \sigma_2(4) = 1$. The Cauchy two-line notation $\sigma = \begin{pmatrix} \underline{\mathbf{u}} & \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{v}} \\ \underline{\mathbf{v}} & \underline{\mathbf{w}} & \underline{\mathbf{v}} & \underline{\mathbf{u}} \end{pmatrix}$ with a duplicated value in the top line does not specify a unique PBP permutation but it does specify a set $\{\sigma_1, \sigma_2\}$ of possible PBP permutations. However this Cauchy two-line notation cannot specify any VBP at all which will perform the desired sequence permutation. If it is claimed that the VBP can simply substitute the above sequence of vectors with $\begin{pmatrix} 1 & 2 & 3 & 4 \end{pmatrix}$ then we are back to working with positions again so this is not value-based after all, it is simply reversing the position mapping of the PBP.

Thus VBP's are not suitable for permuting sequences which can contain duplicate object values but PBP's can handle these without difficulty.

A second advantage of PBP's when working with matrix rows or columns is that the function σ remains a function from $[n]$ to $[n]$, whereas with VBP's the domain of the function σ becomes the set of rows or columns of the particular matrix concerned which is more cumbersome. The PBP always has its associated permutation function σ as a function from $[n]$ to $[n]$ irrespective of the kind of objects being permuted, since σ maps positions to positions, whereas with VBP the domain and range σ changes with the type of objects being permuted even though the permutation as a reordering process is still the same.

A further disadvantage of VBP's is seen when we come to study the group structure of permutations in §5. The VBP approach leads to an anti-isomorphism between PERM(n) and S_n, so that S_n then becomes a permutation 'anti-group' rather than a permutation group, whereas with the PBP approach an isomorphism is produced so that PERM(n) and S_n are then identical as groups (albeit with elements in different categories). This is discussed in §5.1.

Despite the advantages of PBP, in some areas of application or theory the VBP might be considered preferable, each case has to be considered for its own particular requirements. In this document we use PBP only for all the permutation properties developed, but at certain points we discuss aspects of the VBP.

3.1 Translating Between VBP's And PBP's

As we seen above any Cauchy two-line notation specifies a VBP so long as there are not duplicates in the top line. Even when there are duplicates in the top line the Cauchy two-line notation specifies a set of PBP's, which is a set of just one in the case where no duplicates occur. If the top line is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ we can compare the permutation function τ of the VBP with the permutation function σ of the PBP, since each of these functions then has the same domain (and range) $[n]$.

Applying the two-line notation to the VBP, from

$$\tau = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

we have $\tau(r) = a_r \ \forall \ r \in [n]$, by the definition of a VBP, and applying the two-line notation to the PBP, from

$$\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$$

we have $\sigma(a_r) = r, \ \forall \ r \in [n]$, by the definition of a PBP.

Then we have $\sigma(\tau(r)) = r \ \forall \ r \in [n]$ and so as σ and τ are both bijections, $\tau = \sigma^{-1}$.

Thus the permutation functions for the PBP and VBP that are defined by a given two-line notation are inverses of one another. Note this only applies when the top line of the notation is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$, for otherwise the VBP function will have a different domain and range from the PBP function. When the top line is $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ both σ and τ are members of the symmetric group S_n of degree n, and the mapping $\sigma \mapsto \tau$ is just the inverse function mapping $f \mapsto f^{-1}$ on S_n (an anti-automorphism [9] on the group S_n).

Thus there is a simple translation between the PBP and VBP specified by a given Cauchy two-line notation with a top line of $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$. However we still need to make clear which of the two types of permutation we are using in a given situation to avoid confusion.

So given a permutation $P \in \text{PERM}(n)$ we can derive two permutation functions σ and τ in S_n for it, for the PBP and VBP interpretations respectively, and σ and τ will be inverses of one another. Conversely we could start out with an arbitrary function $\sigma \in S_n$ and derive two permutations $P, Q \in \text{PERM}(n)$ from it, by using the PBP and VBP interpretations respectively. We find in §6 below that such P and Q are inverse permutations of one another.

4 Compositions Of Permutations

If we have two permutations P and Q of n objects then we could perform Q first then perform P, so we have carried out two reorderings in succession. The net result is a single permutation of the n objects and we can denote this by $P \circ Q$, the composition of P and Q. This is similar to forming a composition $f \circ g$ of functions f and g by defining $(f \circ g)(x) = f(g(x))$. It is clear that such a composition operation, as in the case of functions, is associative, ie. $P \circ (Q \circ R) = (P \circ Q) \circ R$. Then a composition of m permutations can be written as $P_1 \circ P_2 \circ \ldots \circ P_m$ without requiring brackets.

However permutation composition is not commutative, ie. we may have $P \circ Q \neq Q \circ P$, for example define P to be the swap of the first two elements in a sequence of length 3 and Q to be the swap of the second and third elements, then we have $P \circ Q = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ and $Q \circ P = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$.

If P has permutation function σ and Q has permutation function τ then on applying the combined permutation $P \circ Q$ the element at position r is moved to the final position $\sigma(\tau(r))$ since applying Q first moves it to position $\tau(r)$ from which point it is then moved to position $\sigma(\tau(r))$ by P. Thus the permutation function for $P \circ Q$ is $\sigma \circ \tau$, so composition of permutations is compatible with composition of permutation functions. Due to the associativity this extends to $P_1 \circ P_2 \circ \ldots \circ P_m$ having permutation function $\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_m$.

In practice we will use a product notation for composition of permutations and write $P \circ Q$ as PQ, $\sigma \circ \tau$ as $\sigma\tau$, and $\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_m$ as $\sigma_1\sigma_2 \ldots \sigma_m$, since as we discuss in §5 below $\text{PERM}(n)$ and S_n are groups under their respective composition operations, the group operation being considered multiplicative.

5 Permutations As A Group

In §1 we saw that permutations P of n objects were via their permutation functions σ in a one-to-one correspondence with the symmetric group S_n of degree n, ie. we had a bijective mapping $\theta : \text{PERM}(n) \mapsto S_n$. We didn't show that $\text{PERM}(n)$ had a group structure wrt permutation composition by checking the individual group axioms, however we can show this using S_n and θ as follows.

Generally if G is a group and we have a bijection $\phi : G \mapsto H$ to a set H then G induces a group operation on H given by $ab = \phi(\phi^{-1}(a)\phi^{-1}(b))$ and then ϕ is an isomorphism [1, p104-105] from G to H, so that H is essentially the same group as G. Thus given that S_n is a group (under mapping composition) the bijection $\phi = \theta^{-1}$ induces a group structure on $\text{PERM}(n)$ such that if PQ denotes the product of P and Q under this group structure then $PQ = \theta^{-1}(\theta(P)\theta(Q))$, ie. $PQ = \theta^{-1}(\sigma\tau)$, using the notation of §4. But by the compatibility of the composition operations on $\text{PERM}(n)$ and S_n that we observed in §4, $\theta^{-1}(\sigma\tau)$ is just $P \circ Q$, so the induced group operation PQ from S_n is just the composition operation on $\text{PERM}(n)$, and thus we can just write the composition $P \circ Q$ in the product notation PQ, and use the term 'product of permutations' rather than 'composition of permutations'.

$$\text{group of permutations } P \in \text{PERM}(n) \ \xleftrightarrow{\text{isomorphism } \theta} \ \text{group of permutation functions } \sigma \in S_n \quad \text{(PBP)}$$

$\text{PERM}(n)$ is a set whose elements are our native notion of 'permutation of n objects' as opposed to the mathematical formalism of this notion which is the permutation function $\sigma : [n] \mapsto [n]$, the set of all such permutation functions being the set S_n. $\text{PERM}(n)$ and S_n form groups that are isomorphic via θ, ie. essentially the same group, even though the

elements in those groups are in different categories. These groups are non-abelian since as we saw in §4 permutation composition is non-commutative.

5.1 Group Structure For VBP's

In §3.1 above, we saw that for a given permutation $P \in \text{PERM}(n)$, if σ is the permutation function of P on treating P as a PBP and if τ is the permutation function of P on treating P as a VBP then $\tau = \sigma^{-1}$. Thus the mapping from PERM(n) to S_n which maps P to its VBP permutation function τ is the bijection $\theta' : \text{PERM}(n) \mapsto S_n$ given by $\theta' = i \circ \theta$, where $i : S_n \mapsto S_n$ maps each function in S_n to its inverse. However i is an anti-automorphism [9] on the group S_n, ie. $i(fg) = i(g)i(f)$, so that $\theta'(PQ) = i(\theta(PQ)) = i(\theta(P)\,\theta(Q)) = i(\theta(Q))\,i(\theta(P)) = \theta'(Q)\theta'(P)$, so that θ' is an anti-isomorphism from PERM(n) to S_n.

$$\text{group of permutations } P \in \text{PERM}(n) \xleftrightarrow{\text{anti-isomorphism } \theta'} \text{group of permutation functions } \sigma \in S_n \quad \text{(VBP)}$$

It then follows from Theorem A.2 in Appendix A that θ' is an isomorphism from PERM(n)$^{\text{op}}$ to S_n, where PERM(n)$^{\text{op}}$ denotes the 'opposite group' (or 'anti-group') of PERM(n) (Appendix A and [10]). The opposite group G^{op} of a group $(G, \cdot)$ is the group $(G, *)$ which reverses the group operation, ie, $a * b = ba$.

$$\text{anti-group of permutations } P \in \text{PERM}(n)^{\text{op}} \xleftrightarrow{\text{isomorphism } \theta'} \text{group of permutation functions } \sigma \in S_n \quad \text{(VBP)}$$

In the case of PERM(n) the opposite group PERM(n)$^{\text{op}}$ has group operation $PQ = Q \circ P$, and thus in the composite permutation PQ in the opposite group we apply P first and then apply Q. This is counter to the usual convention for the idea of composition — for example in the symmetric group S_X on a general set X, [8], in the mapping composition fg, g is applied first then f. Some confusion would thus be caused by adopting the opposite convention for the group PERM(n).

Thus in summary if we use the VBP type permutation then the group S_n of permutation functions is isomorphic to the 'anti-group' PERM(n)$^{\text{op}}$ of the group PERM(n) of permutations, via the mapping θ' between the permutations and their permutation functions. One example of the effect of that would be that instead of identifying permutation $P_1 \cdots P_m$ with permutation function $\sigma_1 \cdots \sigma_m$ (where σ_i is permutation function of P_i) we would be identifying $P_1 * \cdots * P_m$ with $\sigma_1 \cdots \sigma_m$ (where * denotes the anti-group operation), ie. $P_m \cdots P_1$ with $\sigma_1 \cdots \sigma_m$ (the '$\cdot$' denoting the original group operation), which would be confusing since if we were to then identify P_i with σ_i as is customary, we would obtain $\sigma_m \cdots \sigma_1$ identified with $\sigma_1 \cdots \sigma_m$, and so clearly the identification process has now broken down.

The PBP works better because with it we obtain that S_n is isomorphic to the group PERM(n) via the mapping θ between the permutations and their permutation functions — we are then able to fully identify the respective elements of the two groups, and any group operations done in one of the groups is reflected fully in the other without any inconsistencies arising.

6 Inverse Permutations

The inverse of a permutation P on n objects is defined to be a permutation P^{-1} which completely undoes the reordering performed by P, ie. the composition of P followed by P^{-1} is the identity permutation. The group structure of PERM(n) and the isomorphism θ between PERM(n) and S_n described in §5 guarantee that :

- an inverse P^{-1} always exists
- P^{-1} is unique
- P^{-1} is both a left and a right inverse, ie. $PP^{-1} = P^{-1}P =$ identity permutation
- the permutation function of P^{-1} is the inverse function σ^{-1} of σ, where σ is the permutation function of P

From the general properties of a group [1, p75], the inverse of a product of permutations $\sigma_1\sigma_2 \ldots \sigma_m$ is given by :

$$(\sigma_1\sigma_2 \ldots \sigma_m)^{-1} \quad = \quad \sigma_m^{-1}\, \sigma_{m-1}^{-1}\, \ldots\, \sigma_1^{-1} \tag{5}$$

In §3.1 we considered mapping a single bijection σ in S_n to two permutations $P, Q \in \text{PERM}(n)$ via the PBP and VBP interpretations respectively. Then the PBP P^{-1} would have permutation function σ^{-1}. The VBP Q would have permutation function σ and thus from §3.1 it would have permutation function σ^{-1} if it was reinterpreted as a PBP. Thus if both P^{-1} and Q are interpreted as PBP's they both have the same permutation function σ^{-1}, and so $Q = P^{-1}$.

So the two permutations P, Q in PERM(n) produced by σ in the above way are inverses of one another. So if someone was working in the group S_n and relating to PERM(n) via the VBP convention then the permutations they are working with would be the inverses of the ones we are considering in this article.

Example 4. Let $P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 5 & 2 & 1 \end{pmatrix}$ and find the inverse permutation of P. Applying equation (2) to obtain σ^{-1} from the second line and then inverting this (by reading from the bottom line up to the top line in numerical order) we obtain : $\sigma(1) = 5, \sigma(2) = 4, \sigma(3) = 2, \sigma(4) = 1, \sigma(5) = 3$. Since σ^{-1} is the permutation function of P^{-1}, applying equation (2) to P^{-1} we then obtain : $P^{-1} = \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma(1) & \sigma(2) & \cdots & \sigma(n) \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 4 & 2 & 1 & 3 \end{pmatrix}$.

□

7 Cyclic Permutations

A cyclic permutation (aka a cycle) of n objects maps a sequence of positions r_i in a cyclic fashion $r_1 \to r_2 \to \ldots \to r_k \to r_1$ ($1 \leq k \leq n$, with each r_i a distinct number in $[n]$) by repeated application of the permutation function σ, whilst leaving all of the positions not in the cycle unchanged, ie. it defines σ by :

1. if $k \geq 2$:
   (a) $\sigma(r_i) = r_{i+1}$ for $i = 1, 2, \ldots, k-1$
   (b) $\sigma(r_k) = r_1$
   (c) $\sigma(r) = r \ \forall \ r \in [n] \setminus \{r_1, r_2, \ldots, r_k\}$
2. if $k = 1$:
   σ = identity permutation

Such a permutation is denoted $(r_1 r_2 \ldots r_k)$ and called a k-cycle. Spaces may be needed in the notation if otherwise confusion would result in distinguishing between the terms, but commas are never used. A 1-cycle comprising only one element is just the identity permutation (in the 1-cycle there is no next element after the first and the first and last element are the same thing). For example $(12 \ldots n) = \begin{pmatrix} 1 & 2 & \ldots & n-1 & n \\ n & 1 & \ldots & n-2 & n-1 \end{pmatrix}$ is an n-cycle which shifts all the elements one place to the right, wrapping around at the end, and $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 2 & 1 & 3 & 5 \end{pmatrix} = (413)$ (note positions 2 and 5 don't change).

Note the notation for a k-cycle ($k \geq 2$) is not unique as the cycle can start at any of its points and generate the same 'orbit', eg. the second of the foregoing examples can equally well be written as (341) or (134), these denote the same cyclic permutation as (413).

For a given n, $(r_1 r_2 \ldots r_k)$ only specifies a valid cyclic permutation of n objects if $1 \leq k \leq n$, with each r_i a distinct number in $[n]$. For example (43723) does not specify a valid cyclic permutation of 8 objects because of the repeated 3 and (216) does not specify a valid cyclic permutation of 5 objects as the 6 is out of range.

Given an arbitrary permutation $\sigma = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}$ we can generate a cycle from any starting point $r_1 \in [n]$ by repeatedly applying the function σ to obtain the sequence of positions : $r_1, \sigma(r_1), \sigma(\sigma(r_1)), \cdots$. This sequence is known as the 'orbit' of the point $r_1 \in [n]$ with respect to the function $\sigma : [n] \mapsto [n]$. Denote this sequence by $r_1, r_2, r_3, \cdots$. Since there are only n positions, at some point we must generate a repeated position. If the first repeat occurs at r_k then it must follow that $r_k = r_1$ since if instead we had (for $k \geq 3$) $r_k \in \{r_2, r_3, \ldots, r_{k-1}\}$, say $r_k = r_i$ where $i \in [2, k-1]$, then we have $\sigma(r_{k-1}) = r_k = r_i = \sigma(r_{i-1})$ and hence as σ is bijective $r_{k-1} = r_{i-1}$, but this implies r_{k-1} was repeated at the earlier value r_{i-1} which is a contradiction since we chose r_k to be the first repeat that occurred.

Thus we have obtained a cyclic permutation, σ_1 say, given by $\sigma_1 = (r_1 r_2 \ldots r_k)$, which maps the positions $r_1, r_2, \ldots, r_k$ in the same way as σ. However $\sigma_1(r)$ is not necessarily the same as $\sigma(r)$ for other positions $r \in [n] \setminus \{r_1, r_2, \ldots, r_k\}$, since σ_1 leaves all of these positions unchanged whilst σ may move them.

If we now choose $s_1 \in [n] \setminus \{r_1, r_2, \ldots, r_k\}$ we can generate another sequence $s_1, \sigma(s_1), \sigma(\sigma(s_1)), \ldots$, which will produce another cyclic permutation $\sigma_2 = (s_1 s_2 \ldots s_l)$ in a similar manner as above. It will map the positions $s_1, s_2, \ldots, s_l$ in the same way as σ, though other positions may be mapped differently than σ. If any of the s_j were in the cycle $(r_1 r_2 \ldots r_k)$ it would imply s_1 must be in this cycle which it can't be since we chose $s_1 \in [n] \setminus \{r_1, r_2, \ldots, r_k\}$. Hence the two cycles

$\sigma_1 = (r_1 r_2 \dots r_k)$ and $\sigma_2 = (s_1 s_2 \dots s_l)$ must be disjoint.

We can continue in this way, next time selecting $t_1 \in [n] \setminus (\{r_1, r_2, \dots, r_k\} \cup \{s_1, s_2, \dots, s_l\})$, to obtain a sequence of mutually disjoint cyclic permutations $\sigma_1, \sigma_2, \sigma_3, \dots$, such that each one individually must equal the function σ on the positions within the cycle itself, as we saw with σ_1 and σ_2 above. Some of the σ_i will be 1-cycles, meaning the starting point for that cycle was a fixed point of σ (ie. unchanged by σ). Since each cycle contains at least one position and the cycles are mutually disjoint, and there are only n positions in total, there must come a point when all the positions are exhausted. Let the sequence of cycles thus generated be $\sigma_1, \sigma_2, \dots, \sigma_m$. It follows that σ must be the product (see §4) of these, ie. $\sigma = \sigma_1 \sigma_2 \cdots \sigma_m$, because any cycle leaves unchanged any position that is not within it and every position in the range $(1, 2, \dots, n)$ is present within exactly one of these m cycles. Furthermore we can reorder the σ_i in any manner and their product will still equal σ, since any two disjoint cycles must commute with one another. Note though that non-disjoint cycles generally don't commute, eg. (12) and (23) for $n = 3$, since $(12)(23) = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$ and $(23)(12) = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}$.

Thus we have shown that any permutation σ of n objects can be partitioned into a product of disjoint cycles. Assuming we do not exclude the 1-cycles from this product (each of which equal the identity permutation) this partitioning is unique, ie. any two such products must comprise the exact same set of cycles, since taking any cycle within one of them a point within it must belong to exactly one of the cycles in the other product and then both these cycles are the σ-orbits of that common point and thus equal. This product of disjoint cycles is called the cyclic decomposition of σ. A given permutation is cyclic if and only if its cyclic decomposition contains a single cycle.

Example 5. Determine the cyclic decomposition of $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 6 & 5 & 1 & 7 & 8 & 3 & 4 & 2 \end{pmatrix}$.

$$\begin{aligned} r_1 = 1 &\implies \text{cycle } \sigma_1 = (136) \\ s_1 = 2 &\implies \text{cycle } \sigma_2 = (285) \\ t_1 = 4 &\implies \text{cycle } \sigma_3 = (47). \end{aligned}$$

Hence the cyclic decomposition of σ is $\sigma = (136)(285)(47)$, and the cycles may be placed in any order.

□

8 Transpositions

A transposition is defined as a 2-cycle, ie. it swaps the positions of two of the objects in the sequence and leaves all of the other objects in their original positions. For example $(25) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 5 & 3 & 4 & 2 \end{pmatrix}$.

A transposition is always its own inverse, ie. as a bijection on $[n]$ it is an 'involution'.

An adjacent transposition (AT) is a transposition which swaps an adjacent pair of positions, ie. it is a 2-cycle comprising two consecutive integers, eg. $(34) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 4 & 3 & 5 \end{pmatrix}$.

By applying a series of AT's we can move a particular object in the sequence to any desired position, leaving all the other objects in the same relative position. If we had an arbitrary permutation $\begin{pmatrix} a_1 & a_2 & \cdots & a_n \end{pmatrix}$ we could achieve it by applying a series of AT's to $\begin{pmatrix} 1 & 2 & \cdots & n \end{pmatrix}$ to put a_1 at the beginning then apply another series of AT's to put a_2 in second place, and so on. Continuing in this way we would eventually have all the a_r in the correct place. Thus any permutation can be expressed as a product of AT's. Note that even though these AT's are cycles this is not the cyclic decomposition of the permutation since the AT's are not disjoint.

Example 6. $P = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 1 & 5 & 3 & 2 \end{pmatrix} \implies P$ is achieved by the following sequence of AT's : (34), (23), (12), (45), (34), (45) $\implies P = (45)(34)(45)(12)(23)(34)$, recalling that in a product PQ of permutations, Q is applied first then P (as with function composition fg). Note these AT's are not disjoint cycles so we can't assume they commute (§7).

□

9 Parity Of A Permutation

The parity of a permutation may be defined in different ways. The definition given here is one of the equivalent definitions provided in [13].

Define the set PAR = {even, odd} and define an operation of addition on PAR to mirror the way addition affects parity on $\mathbb{W} = \{0, 1, 2, \ldots\}$, so that if 'par' denotes the parity function from $\mathbb{W}$ to PAR then we have the additivity property :

$$\text{par}(a + b) = \text{par}(a) + \text{par}(b) \ \forall \ a, b \in \mathbb{W} \tag{6}$$

ie. define '+' on PAR as follows :

$$\begin{aligned} \text{even} + \text{even} &= \text{even} \\ \text{even} + \text{odd} &= \text{odd} \\ \text{odd} + \text{even} &= \text{odd} \\ \text{odd} + \text{odd} &= \text{even} \end{aligned}$$

It is easily checked that '+' on PAR is associative and commutative and hence (PAR, +) forms an abelian group, a cyclic group of order 2 generated by 'odd', with 'even' the identity, each member being its own inverse (ie. an 'involution'), and subtraction being equivalent to addition (ie. $a - b = a + b \ \forall \ a, b \in \text{PAR}$).

In this section we will define a parity function on PERM(n), the set of all permutations of n objects, ie. a function par : PERM(n) $\mapsto$ PAR. Note we use the same function name as for the function par : $\mathbb{W} \mapsto$ PAR, it will be clear from the context which function is intended.

Since we usually refer to members of the set PERM(n) by their associated permutation functions $\sigma \in S_n$, we have a third parity function par : $S_n \mapsto$ PAR which mirrors the parity function on PERM(n).

The following three theorems allow us to consistently define the function par on PERM(n).

Theorem 1. If a product of m AT's is equal to the identity then m is even.

Proof. Firstly note m cannot be 1 since an AT can never be the identity permutation. If m was zero we would have the empty product which is the identity permutation by convention. If $n = 2$ then m is the number of times positions 1 and 2 are swapped which must be even since the final order is the same as the original order. So consider the case $n \geq 3$.

Let the sequence of AT's be $\sigma_m, \sigma_{m-1}, \ldots, \sigma_1$. So under the permutation $\sigma_m \sigma_{m-1} \cdots \sigma_1$ the sequence $(1, 2, 3, \ldots, n)$ is mapped back to $(1, 2, 3, \ldots, n)$.

1	2	.....	n	original sequence
$a_1^{(1)}$	$a_2^{(1)}$	.....	$a_n^{(1)}$	σ_1 applied
$\vdots$	$\vdots$		$\vdots$	$\vdots$
$a_1^{(i)}$	$a_2^{(i)}$	.....	$a_n^{(i)}$	σ_i applied
$\vdots$	$\vdots$		$\vdots$	$\vdots$
$a_1^{(m-1)}$	$a_2^{(m-1)}$	.....	$a_n^{(m-1)}$	σ_{m-1} applied
1	2	.....	n	σ_m applied

Consider a value v in the original sequence. The position of v is potentially changed each time a σ_i is applied, and after σ_m is applied the v is back in its original position. Given any $i \in [1, m]$ either σ_i moves v or it does not, but note this is not purely a function of σ_i, which maps positions only, it depends also on the cumulative effect of $\sigma_1, \ldots, \sigma_{i-1}$ which determines where v is at the point σ_i is applied, ie. we are not asking whether $\sigma(v) \neq v$. If we consider only the σ_i which move v then since each σ_i is an AT, ie. it swaps two adjacent positions, each of these σ_i must either move v one place to the left or move it one place to the right and thus because v returns to its original position at the end there must be as many left moving σ_i's as there are right moving ones, so that the total number $e_i \in \mathbb{W}$ of σ_i's which move v must be even.

Consider now two distinct values v, w in the original sequence. We show that the number $d_{v,w} \in \mathbb{W}$ of σ_i's which move both v and w must be even. If we look at all m of the σ_i's they can be partitioned as follows :

- those which move both v and w
- those which move v but not w
- those which move w but not v
- those which move neither v nor w

Because the σ_i's are AT's only the σ_i's in the first of these categories change the order of v and w — when only one of v and w is moved it must remain to the same side of the other. After application of all m of the σ_i's, v and w are back in their original order, so there must be an even number of σ_i's in the first category, ie. $d_{v,w}$ must be even.

Then denoting set size with the # symbol we have :

$$\begin{aligned} m &= \#\{\sigma_i\text{'s moving 1}\} + \#\{\sigma_i\text{'s not moving 1}\} \\ &= e_1 + \#\{\sigma_i\text{'s moving 2 but not 1}\} + \#\{\sigma_i\text{'s not moving 2 or 1}\} \\ &= e_1 + e_2 - d_{2,1} + \#\{\sigma_i\text{'s moving 3 but not 2 or 1}\} + \#\{\sigma_i\text{'s not moving 3, 2 or 1}\} \\ &= e_1 + e_2 - d_{2,1} + e_3 - d_{3,2} - d_{3,1} + \#\{\sigma_i\text{'s moving 4 but not 3, 2 or 1}\} + \#\{\sigma_i\text{'s not moving 4, 3, 2 or 1}\} \end{aligned}$$

Continuing this process (it terminates after the first line if $n = 3$), since each e_v and $d_{v,w}$ is even we eventually obtain :

$$\begin{aligned} m &= e + \#\{\sigma_i\text{'s not moving } n-2, \ldots, 2 \text{ or } 1\} \\ &= e + \#\{\sigma_i\text{'s moving } n \text{ and } n-1\} \\ &= e + d_{n,n-1} \end{aligned}$$

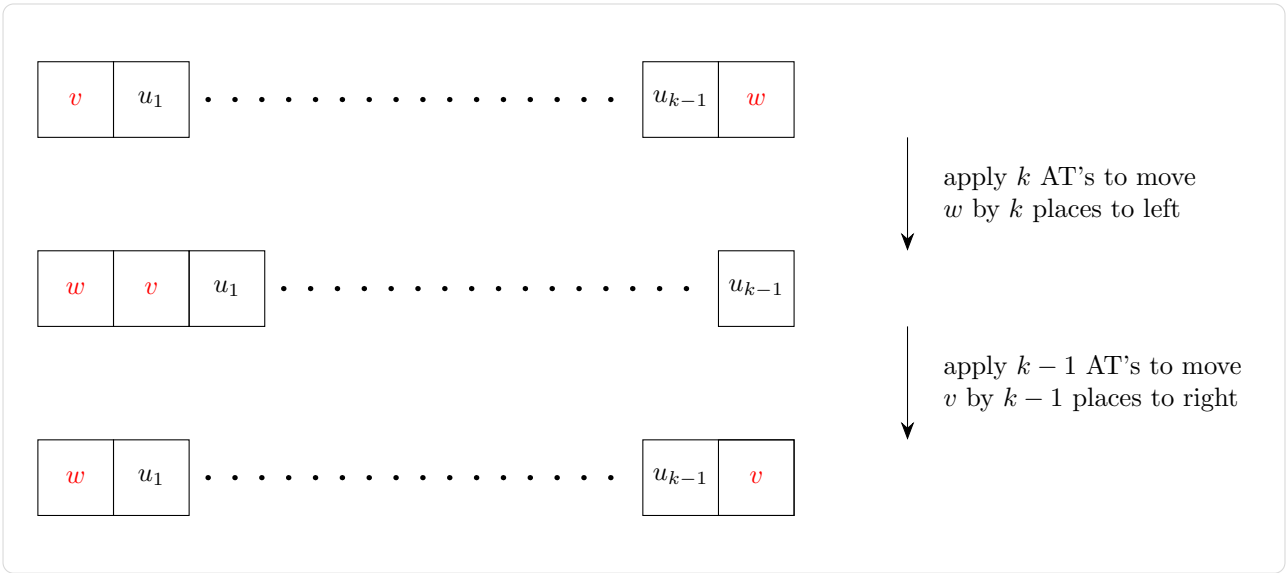
where e is even. Hence m is even, as required.

QED

Theorem 2. Any transposition can be expressed as a product of an odd number of AT's.

Proof. Suppose v and w are the values in the sequence being transposed, with v located k places to the left of w. If $k = 1$ then the transposition is itself an AT. Otherwise consider the case $k > 1$:

Figure 1: Transposition As Odd Number of AT's



As seen in Figure 1 a product of $2k - 1$ AT's applied to the sequence swaps the positions of v and w whilst leaving all other objects in the sequence in their original positions. This product is then equal to the original transposition.

QED

Theorem 3. Suppose a permutation σ can be expressed as a product of m transpositions and as a product of p transpositions, where $m, p \in \mathbb{W}$. Then $\text{par}(m) = \text{par}(p)$.

Proof. Consider the product of m transpositions, $\sigma_1, \sigma_2, \ldots, \sigma_m$, say. Note m could be 0 in which case the product is empty and thus equal to the identity permutation. By Theorem 2, σ_i can be expressed as a product of $e_i + 1$ AT's where $e_i \in \mathbb{W}$ is even. Thus :

$$\begin{aligned} \sigma &= \text{product of } \sum_{i=1}^{m}(e_i + 1) \text{ AT's} \\ &= \text{product of } e + m \text{ AT's, where } e \in \mathbb{W} \text{ is even} \end{aligned}$$

Similarly :

$$\sigma = \text{product of } d + p \text{ AT's, where } d \in \mathbb{W} \text{ is even}$$

By equation (5) the inverse permutation σ^{-1} of the latter product of $d + p$ AT's is also a product of $d + p$ AT's (recall any AT is its own inverse). Then composing this inverse with the former product σ of $e + m$ AT's we obtain a product of $e + m + d + p$ AT's which is equal to the identity permutation. Then :

$$\begin{aligned} \text{par}(e + m + d + p) &= \text{even, by Theorem 1} \\ \Rightarrow \text{par}(e) + \text{par}(m) + \text{par}(d) + \text{par}(p) &= \text{even, using the additivity property (6) of the par function} \\ \Rightarrow \text{par}(m) + \text{par}(p) &= \text{even, since } e \text{ and } d \text{ have even parity and even is the identity} \\ &\quad \text{within the additive group (PAR, +)} \\ \Rightarrow \text{par}(m) &= \text{par}(p), \text{ from the above definition of '+' on PAR} \end{aligned}$$

QED

Theorem 3 guarantees that the following definition of the parity function $\text{par} : \text{PERM}(n) \mapsto \text{PAR}$ is consistent. Recall from §8 that every permutation can be expressed in at least one way as a product of transpositions.

Definition 1. Given a permutation σ of n objects, $\text{par}(\sigma)$ is defined to be the parity of the length $m \in \mathbb{W}$ of any sequence of transpositions whose product equals σ.

□

Note in the above definition, as in other definitions and theorems, we could just as well refer to a 'permutation P' within $\text{PERM}(n)$ instead of a 'permutation σ' within S_n — the two are identified by the group isomorphism θ between $\text{PERM}(n)$ and S_n (§5).

Theorem 4. If σ and τ are permutations of n objects then :

(a) $\text{par}(\sigma\tau) = \text{par}(\sigma) + \text{par}(\tau)$

(b) $\text{par}(\sigma^{-1}) = \text{par}(\sigma)$

Proof. (a) Express σ as a product of m transpositions and τ as a product of p transpositions. Then $\sigma\tau$ is the product of $m + p$ transpositions, so :

$$\begin{aligned} \text{par}(\sigma\tau) &= \text{par}(m + p), \text{ by definition of parity of permutation} \\ &= \text{par}(m) + \text{par}(p), \text{ by additivity property (6) of par function} \\ &= \text{par}(\sigma) + \text{par}(\tau), \text{ by definition of parity of permutation} \end{aligned}$$

(b) We have :

$$\begin{aligned} \text{par}(\sigma\sigma^{-1}) &= \text{par(identity permutation)} \\ \Rightarrow \text{par}(\sigma) + \text{par}(\sigma^{-1}) &= \text{even} \\ \Rightarrow \text{par}(\sigma^{-1}) &= \text{par}(\sigma), \text{ from the above definition of '+' on PAR} \end{aligned}$$

QED

10 Signature Of A Permutation

The signature of a permutation may be defined in different ways. The definition given here is one of the equivalent definitions provided in [14].

Definition 2. The signature $\text{sgn}(\sigma)$ of a permutation σ of n objects is defined by :

$$\text{sgn}(\sigma) \quad = \quad \begin{cases} 1, & \text{if par}(\sigma) \text{ is even} \\ -1, & \text{if par}(\sigma) \text{ is odd} \end{cases}$$

where $\text{par} : S_n \mapsto \text{PAR}$ is the parity function defined in §9 above.

□

Thus if σ is the product of m transpositions we have :

$$\text{sgn}(\sigma) \quad = \quad (-1)^m.$$

Theorem 5. If σ and τ are permutations of n objects then :

(a) $\text{sgn}(\sigma\tau) = \text{sgn}(\sigma) \cdot \text{sgn}(\tau)$

(b) $\text{sgn}(\sigma^{-1}) = \text{sgn}(\sigma)$

Proof. (a) Express σ as a product of m transpositions and τ as a product of p transpositions. Then $\sigma\tau$ is the product of $m + p$ transpositions, so :

$$\begin{aligned} \text{sgn}(\sigma\tau) &= (-1)^{m+p} \\ &= (-1)^m \cdot (-1)^p \\ &= \text{sgn}(\sigma) \cdot \text{sgn}(\tau) \end{aligned}$$

(b) We have :

$$\begin{aligned} \text{sgn}(\sigma\sigma^{-1}) &= \text{sgn(identity permutation)} \\ \Rightarrow \quad \text{sgn}(\sigma) \cdot \text{sgn}(\sigma^{-1}) &= 1 \\ \Rightarrow \quad \text{sgn}(\sigma^{-1}) &= 1/\text{sgn}(\sigma) \\ \Rightarrow \quad \text{sgn}(\sigma^{-1}) &= \text{sgn}(\sigma) \end{aligned}$$

QED

11 Calculating Parity And Signature Of A Function in S_n

If we are given a bijection σ in S_n, by definition its parity and signature are the parity and signature of the underlying PBP permutation $P \in \text{PERM}(n)$ that σ represents. To calculate the parity of P we can first use equation (2) to immediately write down the inverse permutation P^{-1} of P :

$$P^{-1} \quad = \quad \begin{pmatrix} 1 & 2 & \cdots & n \\ \sigma(1) & \sigma(2) & \cdots & \sigma(n) \end{pmatrix}, \tag{7}$$

since P^{-1} has permutation function σ^{-1}. From this two-line notation, which shows the rearrangement that P^{-1} performs, we could then determine a sequence of transpositions which achieves the permutation P^{-1} and then the parity of the length of that sequence will by definition be the parity of P^{-1}. But from Theorem 4(b) this gives us the parity of P. We then determine the signature of P from the parity of P using Definition 2 above, and this is then the required signature of σ.

A question arises of what would happen if for the same bijection $\sigma \in S_n$ someone used the VBP convention to calculate the parity and signature of σ. As we seen in §6 above this would produce a permutation in $\text{PERM}(n)$ that would be the inverse of the one we produced above using the PBP convention for the same σ. Thus by Theorem 4(b) the same parity and signature would result. As an example, in the Leibniz formula for matrix determinants [5] the signature of a $\sigma \in S_n$ is required and in calculating that signature it would not matter whether we used the PBP or the VBP associated with σ, the same signature value of ± 1 would result.

Example 7. Let $\sigma : [5] \mapsto [5]$ be defined by $\sigma(1) = 4, \sigma(2) = 3, \sigma(3) = 1, \sigma(4) = 5, \sigma(5) = 2$ and determine $\text{sgn}(\sigma)$. From equation (2) we have $P^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 4 & 3 & 1 & 5 & 2 \end{pmatrix}$ and so P^{-1} can be achieved by applying the following sequence of AT's (which put 4 to the front, then 3 to the front, and so on, until all the elements end up in the required positions) : (34), (23), (12), (34), (23), (45), so $P^{-1} = (45)(23)(34)(12)(23)(34)$. Then $\text{par}(P) = \text{par}(P^{-1}) = \text{even}$, and hence $\text{sgn}(\sigma) = 1$. □

12 Extending A Permutation

Given a permutation τ of $n-1$ objects we can extend it to a permutation σ_τ of n objects by specifying one of the position mappings of σ_τ (say $\sigma_\tau(r) = s$) and then using τ to permute the remaining $n-1$ objects out of the n. The following result relates the signatures of τ and σ_τ — it is useful in proving that the Leibniz and Laplace forms of the determinant of an $n \times n$ matrix are equal [5], [6], [15, Theorem 32, §12.4].

Theorem 6. For $\tau \in S_{n-1}$ and $r, s \in [1, n]$ define the extension permutation σ_τ of τ in S_n by :

$$\begin{aligned} \sigma_\tau(r) &= s\,, \\ \sigma_\tau \text{ on } [n] \setminus \{r\} &= \text{determined from } \tau \text{ by identifying } [n] \setminus \{r\} \text{ with } [n-1] \text{ in dom } \sigma_\tau \\ &\quad \text{and } [n] \setminus \{s\} \text{ with } [n-1] \text{ in ran } \sigma_\tau. \end{aligned}$$

Then, $\text{sgn}(\sigma_\tau) = (-1)^{r+s}\,\text{sgn}(\tau)$.

Proof. Let $\theta_n : \text{PERM}(n) \mapsto S_n$ (§1 and §5) be the standard group isomorphism, and let $P = \theta_n^{-1}(\sigma_\tau)$ and $Q = \theta_{n-1}^{-1}(\tau)$ be the permutations of n and $n-1$ objects respectively that are associated with σ_τ and τ.

Suppose τ is the product of m transpositions, so that $\text{sgn}(\tau) = (-1)^m$, and the permutation Q is likewise the product of m transpositions. We show that σ_τ is a product of $m+r+s-2$ transpositions and hence that $\text{sgn}(\sigma_\tau) = (-1)^{m+r+s-2} = (-1)^{r+s}\,\text{sgn}(\tau)$, as required.

It is helpful to firstly describe how permutation P can be described in terms of the permutation Q. Referring to Figure 2 below we see that P can be achieved by the following steps :

1. place the object originally at position r into position s
2. join $n-1$ objects over position r
3. apply permutation Q to the $n-1$ objects
4. split the $n-1$ objects over position s

To confirm P has σ_τ as its permutation function we can trace how a split positional coordinate over r in the range $[n] \setminus \{r\}$ is mapped by both P and σ_τ. First it goes to a joined positional coordinate in the range $[n-1]$, then it is mapped by τ (or equivalently Q) to another joined positional coordinate in the range $[n-1]$, and then finally it is mapped to a split positional coordinate over s in the range $[n] \setminus \{s\}$. The final position this gives is the same for both the function σ_τ and the above word description of P.

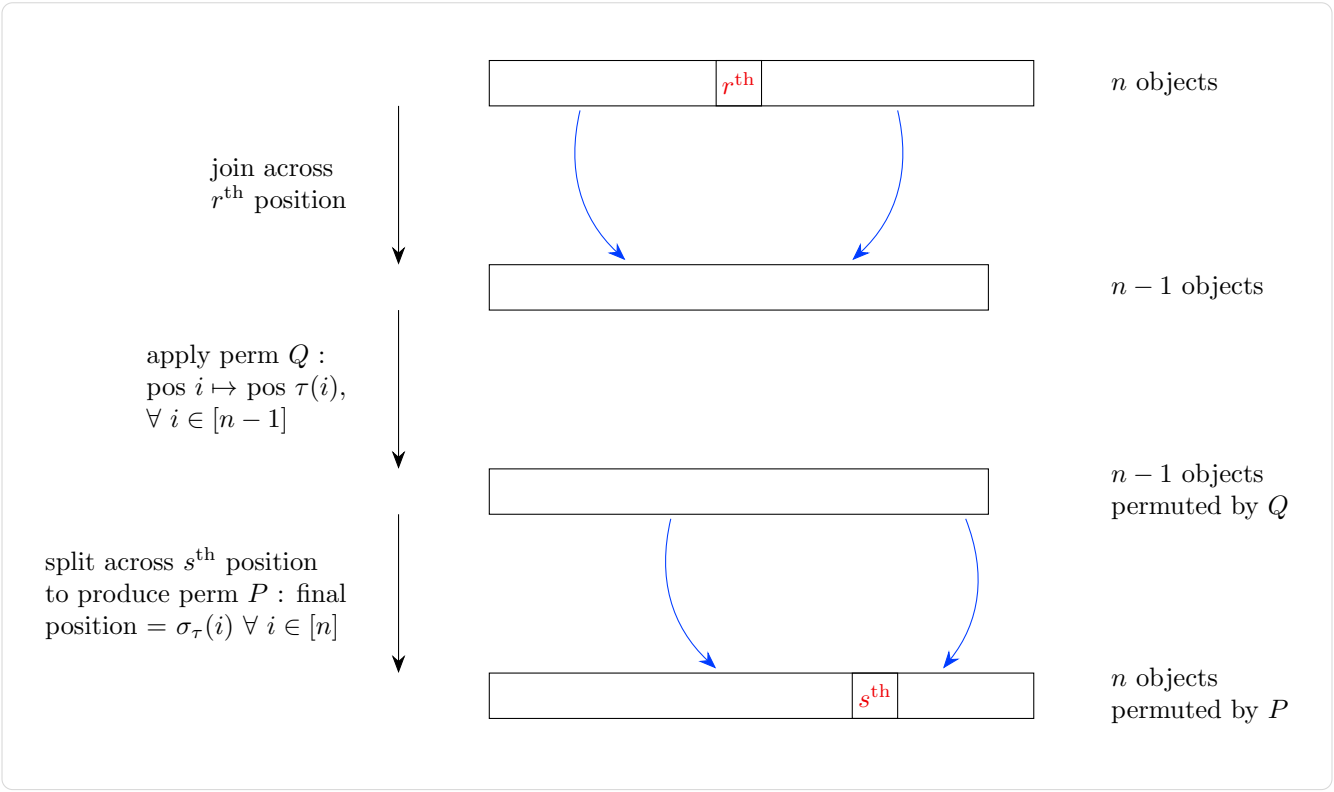
To calculate its signature we require to express P as a product of transpositions, and we can achieve this through a modification of the above procedure :

1. Apply permutation Q (equivalent to m transpositions) to the $n-1$ objects in place within the n objects — note it does not matter that there is a gap at the r^{th} position, we are still just reordering a list of $n-1$ objects. These $n-1$ objects are now in the correct relative order as required by P. This is because in step 4 of the procedure above if we had split the result of Q over the r^{th} object instead of over the s^{th} object then the $n-1$ objects would still be in the correct relative order for P (splitting them over the s^{th} object would put them in the correct absolute order for P).
2. Through a sequence of $r-1$ AT's now move the object at position r to position 1. The relative order of the other $n-1$ objects remains unchanged by this.
3. Through a sequence of $s-1$ AT's now move the object at position 1 to position s. The relative order of the other $n-1$ objects remains unchanged by this.

This puts the object originally at position r into position s as required by P, and it puts the other $n-1$ objects into the correct relative positions for P. It then follows that all of the n objects are now in the correct absolute positions for P. Thus we have achieved the permutation P (and hence σ_τ) by a sequence of $m+r+s-2$ transpositions.

QED

Figure 2: Extending A Permutation From Q To P



13 PERM(n) Versus S_n

Above we developed theorems and definitions concerning the group PERM(n), though for convenience we often referred to a member P of PERM(n) by its associated permutation function σ which is in the group S_n. This can cause confusion. Are we working in PERM(n) or are we working in S_n? The answer is both simultaneously because of the isomorphism θ : PERM(n) $\mapsto$ S_n (§1 and §5) which means that everything we do in the group PERM(n) is exactly mirrored in the group S_n, ie. we are not just using S_n as a convenient naming convention, something is actually happening in the group S_n as a result of what we do in PERM(n) — and vice-versa. This mirroring is broken if we work with VBP's instead of PBP's, since as we seen in §5.1 the one-to-one correspondence between PERM(n) and S_n then changes from isomorphic to anti-isomorphic, since the permutation function becomes σ^{-1} (§3.1).

So all of the above theorems and definitions concerning PERM(n) have their exact counterparts in S_n, ie. the set of all bijections on $[n]$. So for example saying a bijection $\sigma \in S_n$ is a k-cycle means the associated $P \in$ PERM(n) is a k-cycle according to the definition of k-cycle that we made in §7. Saying σ has a certain parity or signature means the underlying permutation within PERM(n) has that parity or signature, as we defined those properties in §9 and §10. In the proof of Theorem 6 above it was helpful to think in terms of permutations rather than in terms of bijections in order to visualize the reordering process.

But since we have adopted the convention of referring to the underlying permutation within PERM(n) by its associated permutation function σ in S_n some confusion can arise. Permutations and bijections are different categories and to conflate these concepts as being the same is a category error. It is not a category error however to use one of the concepts to refer to the other using a convention that is understood. The identification between the groups PERM(n) and S_n is very useful because we can use whichever one is most appropriate in a given situation whilst knowing that anything we do in one is completely mirrored in the other.

Appendix A

Opposite Groups And Anti-Isomorphisms

Definition A.1. Given a group $(G, \cdot)$ the opposite group (aka anti-group) G^{op} of $(G, \cdot)$ [10] is the group $(G, *)$ where the binary operation $*$ is defined by $a * b = ba$.

□

Note that in the above definition the binary operation * satisfies the group axioms, and $a_1 * \cdots * a_n = a_n \cdots a_1 \; \forall \, a_i \in G$, and * has the same group identity and inverses as the original group operation ' $\cdot$ '. It is clear that any group $(G, \cdot)$ is abelian iff it equals its opposite group.

Definition A.2. Let $\theta : G \mapsto H$ be a function from group G to group H. Then :

(a) θ is an anti-homomorphism [9] if $\theta(ab) = \theta(b)\theta(a)$

(b) θ is an anti-isomorphism if it is bijective and an anti-homomorphism

(c) G and H are anti-isomorphic ($G \overset{a}{\sim} H$) if there exists an anti-isomorphism from G to H

(d) θ is an anti-automorphism on G if it is an anti-isomorphism : $G \mapsto G$

□

Example A.1.

(a) The primary example of an anti-isomorphism is the anti-automorphism $\theta : G \mapsto G$ on a general group G defined by $\theta(a) = a^{-1}$, since $\theta(ab) = (ab)^{-1} = b^{-1}a^{-1} = \theta(b)\theta(a)$. Thus for any group G we have $G \overset{a}{\sim} G$.

(b) On the general linear group $\text{GL}(n)$ [11] of non-singular matrices over a general field $\mathbb{F}$, the function θ defined by $\theta(A) = A^T$ (transpose of A) is a bijective map : $\text{GL}(n) \mapsto \text{GL}(n)$ with $\theta(AB) = (AB)^T = B^T A^T = \theta(B)\theta(A)$, and thus θ is an anti-automorphism on $\text{GL}(n)$.

□

Below we use the notation $G \sim H$ to denote that the groups G and H are isomorphic [1, p104-105].

Theorem A.1.

(a) If $\theta : G \mapsto H$ is an anti-isomorphism then $\theta^{-1} : H \mapsto G$ is an anti-isomorphism. Thus $G \overset{a}{\sim} H \Rightarrow H \overset{a}{\sim} G$.

(b) If $\theta : G \mapsto H$ and $\phi : H \mapsto K$ are anti-isomorphisms then $\phi\theta : G \mapsto K$ is an isomorphism. Thus $G \overset{a}{\sim} H$ and $H \overset{a}{\sim} K$ $\Rightarrow G \sim K$.

Proof.

(a) Given any $a, b \in G$:

$$\begin{aligned} & \theta(ab) = \theta(b)\theta(a) \\ \Rightarrow \quad & \text{applying } \theta^{-1},\ ab = \theta^{-1}(\theta(b)\theta(a)) \\ \Rightarrow \quad & \theta^{-1}(\theta(b)\theta(a)) = \theta^{-1}(\theta(a))\theta^{-1}(\theta(b)) \end{aligned}$$

$\Rightarrow \theta^{-1}$ is an anti-isomorphism.

(b) Given any $a, b \in G$:

$$\begin{aligned}(\phi\theta)(ab) &= \phi(\theta(ab)) \\ &= \phi(\theta(b)\theta(a)) \\ &= \phi(\theta(a))\phi(\theta(b)) \\ &= (\phi\theta)(a) \cdot (\phi\theta)(b)\end{aligned}$$

as required.

QED

Theorem A.2. $\theta : G \mapsto H$ is an anti-isomorphism iff $\theta : G^{\text{op}} \mapsto H$ is an isomorphism.

Proof. Given a bijection $\theta : G \mapsto H$:

$$\begin{aligned}\theta \text{ is an anti-isomorphism} &\iff \theta(ab) = \theta(b)\theta(a),\ \forall\ a, b \in G \\ &\iff \theta(b * a) = \theta(b)\theta(a),\ \forall\ a, b \in G \\ &\iff \theta : G^{\text{op}} \mapsto H \text{ is an isomorphism}\end{aligned}$$

as required.

QED

Corollary. For any group $(G, \cdot)$, the function $\theta : G \mapsto G^{\text{op}}$ defined by $\theta(a) = a^{-1}$ is an isomorphism from G to G^{op}. Hence $G \sim G^{\text{op}}$.

Proof. From Example A.1(a), the function θ is an anti-isomorphism from G to G. Then by Theorem A.2, this implies $\theta : G^{\text{op}} \mapsto G$ is an isomorphism. Thus $\theta^{-1} : G \mapsto G^{\text{op}}$ is an isomorphism. But $\theta^{-1} = \theta$, hence

QED

References

All web links retrieved on 27th January 2026.

[1] Thomas A. Whitelaw (1978), *An Introduction To Abstract Algebra*, Blackie & Sons Ltd.

[2] Paul B. Yale (1968), *Geometry And Symmetry*, Holden Day.

[3] Robert G. Bartle & Donald R. Sherbert (1982), *Introduction to Real Analysis*, John Wiley & Sons.

[4] Gowers's Weblog : *Permutations*, https://gowers.wordpress.com/2011/10/16/permutations/

[5] *Leibniz formula for determinants*, https://en.wikipedia.org/wiki/Leibniz_formula_for_determinants

[6] *Laplace expansion*, https://en.wikipedia.org/wiki/Laplace_expansion

[7] *Symmetric group*, https://en.wikipedia.org/wiki/Symmetric_group

[8] *Symmetric group/Multiplication*, https://en.wikipedia.org/wiki/Symmetric_group#Multiplication

[9] *Anti-homomorphism*, https://en.wikipedia.org/wiki/Antihomomorphism

[10] *Opposite group*, https://en.wikipedia.org/wiki/Opposite_group

[11] *General linear group*, https://en.wikipedia.org/wiki/General_linear_group

[12] *Permutation/Two-line notation*, https://en.wikipedia.org/wiki/Permutation#Two-line_notation

[13] *Parity of a permutation/Equivalence of the two definitions*, https://en.wikipedia.org/wiki/Parity_of_a_permutation#Equivalence_of_the_two_definitions

[14] *Parity of a permutation*, https://en.wikipedia.org/wiki/Parity_of_a_permutation

[15] Ross Ure Anderson (2026), *Matrix Inverses: From Geometry To Permutations*, https://archive.org/details/matrix-inverses and https://www.academia.edu/168691367/Matrix_Inverses

[16] Ross Ure Anderson (https://math.stackexchange.com/users/417024/ross-ure-anderson), Mathematics Stack-Exchange, answer to *Cauchy-Binet Formula Proof Intuition (Determinants)*, https://math.stackexchange.com/a/4014267/417024.